What is claimed is:

1. (Currently Amended) A ~~method~~ system, comprising a processor and a memory, the memory storing a set of instructions executable by the processor to:

(1) ~~determining~~ determine a set of present components assigned to a domain, each of the set of present components includes a set of modules wherein the domain is part of a hierarchical set of domains connected by a domain link path;

(2) ~~determining~~ determine a set of symbols imported by the set of modules assigned to the domain, wherein each symbol identifies a memory location storing one of a code and a data structure;

(3) ~~determining~~ determine zero or more needed components to which the domain does not have access and at least one of provides the set of symbols imported by the set of modules, and specified as required by the set of present components; and

(4) ~~adding~~ add the zero or more needed components into one of the domain and the hierarchical domains.

2. (Currently Amended) The ~~method~~ system of claim 1, wherein determining the zero or more needed components includes determining zero or more non-resident needed components which are the zero or more needed components that cannot reside in the domain, as specified by a user of a configuration tool, and determining zero or more resident needed components which are the zero or more needed components that can reside in the domain, as specified by the user.

3. (Currently Amended) The ~~method~~ system of claim 1, wherein the domain has access only to the set of present components assigned to the domain.

4. (Currently Amended) The ~~method~~ system of claim 2, wherein adding the zero or more needed components into the domain includes displaying the zero or more resident needed components to a user of a project facility and allowing the user to add the zero or more needed components into the domain.

5. (Currently Amended) The ~~method~~ system of claim 2, further comprising displaying the zero or more non-resident needed components to a user of a project facility.

6. (Cancelled)

7. (Currently Amended) The ~~method~~ system of claim ~~6~~ 1, wherein a kernel domain is at ~~the~~ a lowest hierarchical level of the domain link path and application domains are at ~~the~~ a highest hierarchical level of the domain link path.

8. (Currently Amended) The ~~method~~ system of claim 7, wherein the particular one of the set of domains does not have access to a set of components not assigned to the particular one of the set of domains, and that provides the set of symbols imported by the set of modules but the particular one of the set of domains is not given entry points to those symbols by the set of domains at lower hierarchical levels in the domain link path, and that are not in at least one of the

zero or more resident needed components, and the zero or more non-resident needed components of any one of the set of domains at lower hierarchical levels in the domain link path, if such levels exist.

9. (Currently Amended) The ~~method~~ system of claim 8, wherein for each of the set of domains in the domain link path, starting at the lowest hierarchical level of the domain link path and traversing to the highest hierarchical level, performing steps (1) to (3) of claim 1 for each of the set of domains.

10. (Currently Amended) The ~~method~~ system of claim 9, further comprising, for each of the set of domains in the domain link path and starting at the highest hierarchical level of that domain link path and traversing to the lowest hierarchical level,

determining the set of symbols imported by the zero or more resident needed components of that domain and determining zero or more second pass needed components that at least one of provides the set of symbols imported by the zero or more resident needed components and to which this domain does not have access, and specified by the component description file as required by the zero or more resident needed components;

inserting the zero or more second pass needed components that can reside in this domain into the zero or more resident needed components for that domain, and

for each of the zero or more second pass needed components that cannot reside in this domain, traversing down the domain link path to the lowest hierarchical level until the particular one of the zero or more second pass needed components can reside in a particular one of the set

of domains in the domain link path and then inserting the particular one of the zero or more second pass needed components into the particular one of the set of domains, otherwise, if the particular one of the zero or more second pass needed components cannot reside in any of the domains of the domain link path, then inserting the particular one of the zero or more second pass needed components into zero or more error components.

11. (Currently Amended) The ~~method~~ system of claim 10, wherein adding the zero or more needed components includes displaying the zero or more resident needed components of a particular one of the set of domains to a user of a project facility and allowing the user to add the zero or more resident needed components into that domain.

12. (Currently Amended) The ~~method~~ system of claim 10, further comprising, displaying the zero or more non-resident needed components of a particular one of the set of domains to a user of a project facility.

13. (Currently Amended) A system implemented by executing a set of instructions on a processor, comprising:

a first parser that lists a set of symbols and a set of modules that imports or exports the set of symbols;

a second parser that maps the set of modules to a set of components and specifies dependencies among the set of components; and

a project analysis utility that

(1) determines a set of present components assigned to a domain, each of the set of present components includes a subset of the set of modules wherein the domain is part of a hierarchical set of domains connected by a domain link path;

(2) determines a subset of the set of symbols imported by the subset of modules assigned to the domain, wherein each symbol identifies a memory location storing one of a code and a data structure;

(3) determines zero or more needed components to which the domain does not have access and at least one of provides the subset of symbols imported by the subset of modules, and specified in the mapping by the second parser as required by the set of present components; and

(4) adds the zero or more needed components into one of the domain and the hierarchical domains.

14. (Original) The system of claim 13, further comprising a configuration tool, coupled to the project analysis utility, at least one of creates the domain, assigns the set of present components to the domain, and specifies whether a particular one of the set of components may reside in the domain.

15. (Original) The system of claim 13, wherein the project analysis utility determines zero or more non-resident needed components which are the zero or more needed components that cannot reside in the domain, as specified by a user of the configuration tool, and determines zero or more resident needed components which are the zero or more needed components that can

reside in the domain, as specified by the user.

16. (Original) The system of claim 15, wherein the domain has access only to the set of present components assigned to the domain.

17. (Original) The system of claim 15, wherein the project analysis utility displays the zero or more resident needed components to a user of a project facility using a graphics user interface and allows the user to add the zero or more needed components into the domain.

18. (Original) The system of claim 15, wherein the project analysis utility displays the zero or more non-resident needed components to a user of a project facility using a graphics user interface.

19. (Cancelled)

20. (Currently Amended) The system of claim ~~19~~ 13, wherein a kernel domain is at ~~the~~ a lowest hierarchical level of the domain link path and application domains are at ~~the~~ a highest hierarchical level of the domain link path.

21. (Original) The system of claim 20, wherein the particular one of the set of domains does not have access to a set of components not assigned to the particular one of the set of domains, and that provides the set of symbols imported by the set of modules but the particular one of the

set of domains is not given entry points to those symbols by the set of domains at lower hierarchical levels in the domain link path, and that are not in at least one of the zero or more resident needed components, and the zero or more non-resident needed components of any one of the set of domains at lower hierarchical levels in the domain link path, if such levels exist.

22. (Original) The system of claim 21, wherein the project analysis utility, for each of the set of domains in the domain link path, starting at the lowest hierarchical level of the domain link path and traversing to the highest hierarchical level, performs (1) to (3) of claim 13 for each of the set of domains.

23. (Original) The system of claim 22, wherein the project analysis utility, for each of the set of domains in the domain link path and starting at the highest hierarchical level of that domain link path and traversing to the lowest hierarchical level,

determines the set of symbols imported by the zero or more resident needed components of that domain and determines zero or more second pass needed components that at least one of provides the set of symbols imported by the zero or more resident needed components and to which this domain does not have access, and specified in the mapping by the second parser as required by the zero or more resident needed components;

inserts the zero or more second pass needed components that can reside in this domain into the zero or more resident needed components for that domain; and

for each of the zero or more second pass needed components that cannot reside in this domain, traverses down the domain link path to the lowest hierarchical level until the particular

one of the zero or more second pass needed components can reside in a particular one of the set of domains in the domain link path and then inserts the particular one of the zero or more second pass needed components into the particular one of the set of domains, otherwise, if the particular one of the zero or more second pass needed components cannot reside in any of the domains of the domain link path, then inserts the particular one of the zero or more second pass needed components into zero or more error components.

24. (Original) The system of claim 23, wherein the project analysis utility, displays the zero or more resident needed components of a particular one of the set of domains to a user of a project facility using a graphics user interface and allows the user to add the zero or more resident needed components into that domain.

25. (Original) The system of claim 23, wherein the project analysis utility displays the zero or more non-resident needed components of a particular one of the set of domains to a user of a project facility using a graphics user interface.

26. (Currently Amended) A ~~method~~ system, comprising a processor and a memory, the memory storing a set of instructions executable by the processor to:

(1) ~~determining~~ determine a set of present components assigned to a domain;

(2) ~~determining~~ determine zero or more precious components specified by a user of a configuration tool as not removable from the domain, each of the zero or more precious components includes a set of modules;

(3) ~~determining~~ determine a set of symbols imported by the set of modules in each of the zero or more precious components, wherein each symbol identifies a memory location storing one of a code and a data structure;

(4) ~~determining~~ determine zero or more needed components to which the domain does not have access and at least one of provides the set of symbols imported by the set of modules, and specified as required by the zero or more precious components;

(5) if one or more of the zero or more needed components is found in the set of present components, then ~~moving~~ move the one or more of the set of present components into the zero or more precious components; and

(6) ~~removing~~ remove the set of present components from the domain.

27. (Currently Amended) The ~~method~~ system of claim 26, wherein the domain has access only to the set of present components assigned to the domain.

28. (Currently Amended) The ~~method~~ system of claim 27, wherein removing the set of present components from the domain includes displaying the set of present components to a user of a project facility and allowing the user to remove the set of present components from the domain.

29. (Currently Amended) The ~~method~~ system of claim 26, wherein the domain is part of a set of domains and a domain link path connects the set of domains between a highest hierarchical level of the domain link path and a lowest hierarchical level.

30. (Currently Amended) The ~~method~~ system of claim 29, wherein a kernel domain is at the lowest hierarchical level of the domain link path and application domains are at the highest hierarchical level of the domain link path.

31. (Currently Amended) The ~~method~~ system of claim 30, wherein the particular one of the set of domains does not have access to a set of components not assigned to the particular one of the set of domains, and that provides the set of symbols imported by the set of modules but the particular one of the set of domains is not given entry points to those symbols by the set of domains at lower hierarchical levels in the domain link path.

32. (Currently Amended) The ~~method~~ system of claim 31, wherein for each of the set of domains in the domain link path, starting at the highest hierarchical level of the domain link path and traversing to the lowest hierarchical level, performing steps (1) to (5) of claim 28 for each of the set of domains.

33. (Currently Amended) The ~~method~~ system of claim 32, wherein moving the one or more of the set of present components into the zero or more precious components includes, for each of the zero or more needed components of a particular one of the set of domains in the domain link path, starting with that domain and traversing down the domain link path toward the lowest hierarchical level until the particular one of the zero or more needed components is equal to a particular one of the set of present components of one of the set of domains in the domain link path and then moving the particular one of the set of present components to the zero or more

precious components of the particular one of the set of domains.

34. (Currently Amended) The ~~method~~ system of claim 33, wherein removing the set of present components from the domain includes displaying the set of present components of a particular one of the set of domains to a user of a project facility, and allowing the user to remove one or more of the set of present components.

35. (Currently Amended) A system implemented by executing a set of instructions on a processor, comprising:

a first parser that lists a set of symbols and a set of modules that imports or exports the set of symbols;

a second parser that maps the set of modules to a set of components and specifies dependencies among the set of components; and

a project analysis utility that

(1) determines a set of present components assigned to a domain;

(2) determines zero or more precious components specified by a user of a configuration tool as not removable from the domain, each of the zero or more precious components includes a subset of the set of modules;

(3) determines a subset of the set of symbols imported by the subset of modules, wherein each symbol identifies a memory location storing one of a code and a data structure;

(4) determines zero or more needed components to which the domain does

not have access and at least one of provides the subset of symbols imported by the subset of modules, and specified as required by the zero or more precious components;

(5) if one or more of the zero or more needed components is found in the set of present components, then moves the one or more of the set of present components into the zero or more precious components; and

(6) removes the set of present components from the domain.

36. (Original) The system of claim 35, wherein the configuration tool, coupled to the project analysis utility, at least one of creates the domain, assigns the set of present components to the domain, and designates the zero or more precious components in the domain.

37. (Original) The system of claim 35, wherein the domain has access only to the set of present components assigned to the domain.

38. (Original) The system of claim 37, wherein the project analysis utility displays the set of present components to a user of a project facility using a graphics user interface and allows the user to remove the set of present components from the domain.

39. (Original) The system of claim 38, wherein the domain is part of a set of domains and a domain link path connects the set of domains between a highest hierarchical level of the domain link path and a lowest hierarchical level.

40. (Original) The system of claim 39, wherein a kernel domain is at the lowest hierarchical level of the domain link path and application domains are at the highest hierarchical level of the domain link path.

41. (Original) The system of claim 40, wherein the particular one of the set of domains does not have access to a set of components not assigned to the particular one of the set of domains, and that provides the set of symbols imported by the set of modules but the particular one of the set of domains is not given entry points to those symbols by the set of domains at lower hierarchical levels in the domain link path.

42. (Original) The system of claim 41, wherein the project analysis utility, for each of the set of domains in the domain link path, starting at the highest hierarchical level of the domain link path and traversing to the lowest hierarchical level, performing (1) to (5) of claim 37 for each of the set of domains.

43. (Original) The system of claim 42, wherein the project analysis utility, in order to move the one or more of the set of present components into the zero or more precious components, for each of the zero or more needed components of a particular one of the set of domains in the domain link path, starting with that domain and traversing down the domain link path to the lowest hierarchical level until the particular one of the zero or more needed components is equal to a particular one of the set of present components of one of the set of domains in the domain link path and then move the particular one of the set of present components to the zero or more

precious components of the particular one of the set of domains.

44. (Original) The system of claim 43, wherein the project analysis utility removes the set of present components by displaying the set of present components of a particular one of the set of domains to a user of a project facility using a graphics user interface, and by allowing the user to remove one or more of the set of present components.

45. (Currently Amended) A device comprising:

a medium; and

a set of instructions recorded on the medium;

wherein the set of instructions, when executed by a processor, cause the processor to:

(1) determine a set of present components assigned to a domain, each of the set of present components includes a set of modules wherein the domain is part of a hierarchical set of domains connected by a domain link path;

(2) determine a set of symbols imported by the set of modules assigned to the domain, wherein each symbol identifies a memory location storing one of a code and a data structure;

(3) determine zero or more needed components to which the domain does not have access and at least one of provides the set of symbols imported by the set of modules, and specified as required by the set of present components; and

(4) add the zero or more needed components into one of the domain and the hierarchical domains.

46. (Currently Amended) A device comprising:

a medium; and

a set of instructions recorded on the medium;

wherein the set of instructions, when executed by a processor, cause the processor to:

(1) determine a set of present components assigned to a domain;

(2) determine zero or more precious components specified by a user of a project facility as not removable from the domain, each of the zero or more precious components includes a set of modules;

(3) determine a set of symbols imported by the set of modules in each of the zero or more precious components, wherein each symbol identifies a memory location storing one of a code and a data structure;

(4) determine zero or more needed components to which the domain does not have access and at least one of provides the set of symbols imported by the set of modules, and specified as required by the zero or more precious components;

(5) if one or more of the zero or more needed components is found in the set of present components, then move the one or more of the set of present components into the zero or more precious components; and

(6) remove the set of present components from the domain.

47. (Currently Amended) The ~~method~~ system of claim 1, wherein the domain is an operation system protection domain.

48. (Currently Amended) The ~~method~~ system of claim 1, further comprising:

creating a bootable application including the domain.

49. (Currently Amended) The ~~method~~ system of claim 48, further comprising:

loading the bootable application on a target processor.

50. (Currently Amended) The ~~method~~ system of claim 1, wherein the set of symbols includes an entry point for a second domain.

51. (Currently Amended) The ~~method~~ system of claim 50, wherein the second domain provides access to a resource not directly accessible by the domain.

52. (Previously Presented) The system of claim 13, wherein the domain is an operating system protection domain.

53. (Previously Presented) The system of claim 13, further comprising:

a bootable application including the domain and all the need components.

54. (Previously Presented) The system of claim 13, wherein the set of symbols imported by the subset of modules assigned to the domain includes an entry point to a second domain.

55. (Previously Presented) The system of claim 54, wherein the second domain provides access to a resource not directly accessible by the domain.